模拟集成电路系统级建模方法研究*

路卢
中国科学院自动化研究所国家专用集成电路设计工程技术研究中心
北京，100190
lu.lu@ia.ac.cn†

摘要：随着工艺进步和需求变革，模拟集成电路设计面临一系列挑战。其中，模拟集成电路在设计中需要满足的性能指标的数量和需要平衡的非理想因素的数量迅速增加是一个重要而紧迫的问题。单纯采用面向仿真的模型难以有效应对设计探索和仿真调度的需求。需要在系统级建模以综合处理性能指标和设计约束条件，并采用敏捷设计方法提高模拟集成电路设计工作的效率。为提升模拟集成电路设计的自动化和智能化水平，以电路模型与性能指标计算解耦分离的框架为基础，本文提出可以处理复杂约束条件的模拟电路系统级建模方法。该方法给出适用于模拟集成电路系统级建模的领域专用语言。该语言包含系统级模型建模所需的语法要素和语法规约。针对模拟集成电路的特点，该方法给出建模语言要素的语义同电路特征的对应关系，并给出将模拟电路元件、拓扑结构以及设计约束视作系统中组件以实现解耦的建模方法。上述方法所得模型以图为基本数据结构并支持图计算。根据上述成果，本文给出包含领域专用语言解析编译、模型存储、图计算等功能在内的系统级建模工具。并给出建模案例表明该建模方法的可用性和适用性。本文工作表明，系统级建模为解决模拟集成电路设计中复杂约束条件解耦提供一个可行的解决思路，具有重要研究价值。
关键词：系统级模型；模拟集成电路；建模方法；领域专用语言；图
DOI:
文章分类:

1 引言

模拟集成电路的构成基础是一系列元件。这些元件组成电路拓扑结构，可以完成特定的功能。在电路模型中，电路被适当抽象简化，并根据模型用途保留必要的信息。

在模拟集成电路设计和仿真中，电路模型既可以用于对目标电路特征的描述与仿真，也可以用于构建目标电路的设计规格说明。研究模拟集成电路建模，对提高模拟集成电路自动化分析的效率具有重要意义。

建模是获得模型的一系列工作的总和。从系统观点来看，模型也是一个系统。[1] 虽然针对同一个目标电路系统建模，但是不同用途的模型在内容和使用方面是不同的。不过，针对同一事物的不同模型之间是可以相互转化的。

目前，模拟集成电路模型对于复杂约束条件缺乏有效整合手段，仿真计算流程自动化调度程度低。系统级与电路结构级模型之间缺乏有效映射，混合层次电路模型的仿真难以兼顾速度与精度的双重要求。建模方法无法满足复杂约束的仿真需求，建模方法敏捷化实践、自动化、智能化程度不高。

*本文系国家自然科学基金项目 (项目号：62141416) 的研究成果之一。
†通讯作者：路卢 (E-mail: lu.lu@ia.ac.cn)

因此，迫切需要从构建复杂约束条件的系统级建模方法角度，对模拟电路系统级建模中的电路构建和复杂约束条件的生成与优化方法进行研究，提出可以处理复杂约束条件的模拟电路系统级建模方法，提升模拟电路设计的自动化和智能化程度。

本文依据系统级建模方法，在模型与算法分离的框架的基础上，提出模拟集成电路的系统级建模语言，建模方法和建模工具。本文提出一种模拟电路设计过程中使用的系统级模型的建模方法。具体来说，本文首先就模拟集成电路建模的现状进行分析，并阐明系统级模型的需求，然后给出一个用于构建模拟集成电路系统的系统级模型的DSL (Domain Specific Language, 领域专门语言)然后给出针对模拟集成电路的建模方法，并就建模工具的功能和特征进行说明。然后以一个差分放大器电路设计为例说明该模型的特点和使用方法。最后总结全文。

2 模拟集成电路建模现状

在电路设计中，模型可以依据其抽象层次分为：系统级模型，行为级模型，RTL (Register Transfer Level, 寄存器传输级)模型和电路级模型等不同层次的模型。系统级模型可以表示有关目标电路系统的组成和约束关系等特征，也可以表示实现目标电路系统设计的规格说明。行为级模型描述的是系统组件的行为特征，它通常是更广泛的系统级模型的一个组成部分。而RTL模型，电路级模型等抽象层级的模型与电路的具体实现有关。

在模拟电路设计领域中，现有多数模型在设定组件时以电路拓扑作为组件构建限定条件，有关参数必须有物理意义限制，受到已有物理公式限制。

系统级模型是一种形式化的目标电路系统描述方式，可以通过转码向多种仿真平台映射可仿真执行的代码。转码时需要重新对参数和物理元件进行绑定。系统级模型通过一系列元素保留并更新这些领域特定知识，依据这些证据和知识更新其模型，并用于后续电路设计和综合[2]。

模拟集成电路建模与仿真是模拟电路设计中密不可分又相互约束的两个重要的工作。长期以来，电路仿真的需求推动了模拟集成电路建模和模型的发展。历经数十年的发展，仿真器逐渐形成一定的体系，如1所示。

许多模拟集成电路建模所得的模型用于仿真获取相应的电路性能指标，它们对电路仿真的精度和速度有重要影响。从面向仿真的建模角度而言，模拟集成电路建模意指对于基本元器件的器件级关系的综合性模型，例如，常见的 BSIM6 模型[3],PSP 模型[4],MVS（MIT Virtual Source model）模型[5],MOS11 模型[6]等。使用这样的仿真系统获取电路性能指标的基本流程可以概括为如图2所示的基本流程。

目前，模拟电路仿真主要采用以 UC Berleley 提出的 SPICE[7-8]及类似的 ngspice[9]等仿真系统为基础的数值化仿真工具，如 Cadence 公司的 Spectre 以及 Synopsys 公司的 HSPICE 仿真器等。数值仿真器以其稳定高效的数值求解结果成为业界的常用仿真工具，是模拟电路仿真的事实上的标准。

虽然随着集成电路工艺特征尺寸的不断减小，器件参数呈现出非离散分布

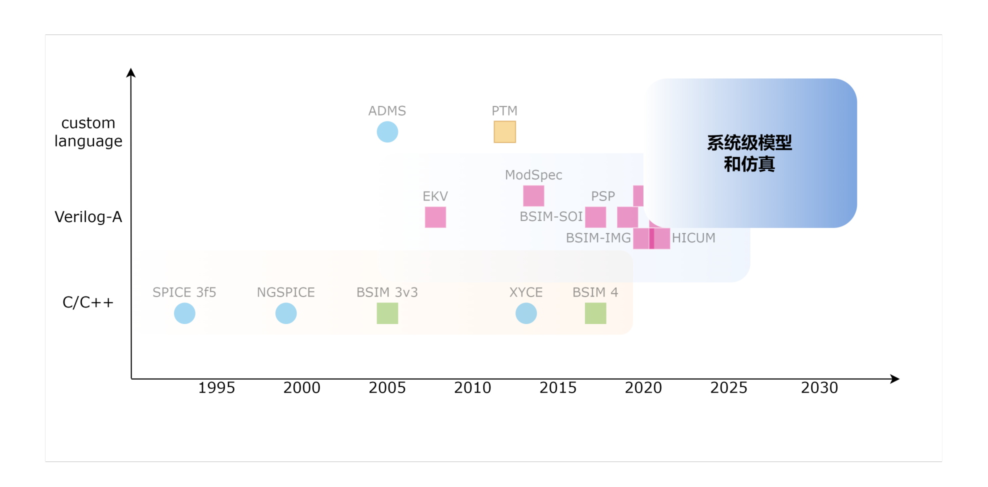


图 1: 模拟集成电路仿真模型的演变

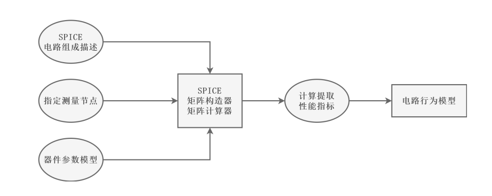


图 2: 传统基于 SPICE 仿真的性能指标获取流程

的特征，且参数的动态范围有所扩大，从仿真角度而言，上述流程和相应的模型能够涵盖模拟集成电路运行的多种特征，有利于准确获取电路的性能指标。另一方面，从模拟集成电路设计的角度而言，在这个流程中，仿真结果与电路元件参数的取值的对应关系不明显，很难从电路的仿真结果获得对于对电路元件的取值的指引。这导致经验不丰富的设计人员，在不确定电路元件与性能指标确切关系的情况下，只能对电路元件取值进行盲目的调整，以期达到所要求的性能指标。

这实质上是用于仿真的电路模型与用于设计的电路模型在内涵上的区别造成的。概括来说，模型使用目的不同，模型涵盖的内容不同，模型的使用方式也不同。当电路模型用于电路特征的描述与仿真时，对于模型的要求是尽可能全面准确反映目标电路的特征。

对于由基本器件构成的功能电路，可以在系统层依据 SPICE 仿真获得的参数进行二次建模，着重于输入输出端口间的数量关系以及运行条件参数对其的影响。相比于器件级模型，就模拟电路系统级建模而言，目前此方面的研究进展相对缓慢，可选择的建模方法相对较少。SPICE-in-the-loop 方法[10-11]，以及 g_m/I_D 方法[12-13] 都是在系统层面进行建模的有益尝试。它们都基于以 SPICE 为核心的电路基本元件的多参数大规模仿真，以此构建查找表来减少系统级仿真时获得性能指标过程中的变量计算负担。此类方法本质上是对于 SPICE 仿真结果的数据拟合和简化，对于加速仿真计算有相对较好的作用。但是受限于 SPICE 仿真获得的数据来源，模型的可扩展性较差。

从模拟集成电路设计角度来说，模拟集成电路性能指标对工作温度、电源、频率以及工艺参数等波动敏感，导致模拟集成电路的性能指标矩阵和环境参数矩阵深度耦合。随着芯片整体结构的日益复杂，模块数量不断增加，模拟电路模块需要满足的性能指标的数量增加迅速，往往导致了性能指标矩阵维度高。此外，模拟电路模块的设计需要处理越来越多的非理想因素对于性能指标的约束和影响，比如各种非线性因素，器件的失配和失调，电路版图带来的附加寄生器件和效应等。在高速与低功耗的总的设计目标之下，任何微小的参数抖动都可能造成电路性能的巨大波动。由于电路规模扩大且约束条件复杂，往往需要在仿真精度和仿真速度方面取得折中。因此，需要从系统层面对模拟电路模块进行建模，探索其性能指标的变动趋势和变动范围，以此为基础支持后续的设计结构优化和性能指标调优。

面对利用器件级建模仿真指导电路设计遇到的现实难题，一种解决方法是引入行为级建模和仿真。行为级建模是一种高层抽象的方法，利用传输函数构造电路结构的宏模型。基于函数的建模和仿真在多物理领域仿真中已经使用多年，例如 ANSYS，MAPLE，modelica[1] 等都是非常优秀的仿真系统，支持行为级建模和仿真。对于模拟集成电路来说，行为级模型给出的函数是一种近似基础上的解析方法，其所得计算模型的使用前提是目标电路结构不发生改变，但电路结构中的参数可以发生改变。某些组件参数的局部线性化，不会影响整体模型得到的性能指标的精确度。而且，这些参数多数为物理组件的自身参数。例如，符号化仿真方法[14-15] 会直接建立小信号电路传输函数与电路元件之间的表达式，然后通过所求的电路传输函数表达式进行求解。行为级模型首先需要获得电路拓扑的某种划分，在划分出的电路单元上设置感兴趣的参数和关系函数，大多缺少对于跨组件物理边界的约束条件的适当的表示方法，且基于电路划分设置的参数和行为模型在系统层不一定完全彼此独立，即在其行为级模型中依然会造成约束条件在多个部件之间耦合。而且在电路设计中，电路结构和参数几乎不可避免发生改变以进行设计探索，因此行为级模型在不同电路结构和参数条件下的适用性并不高。

另一方面，仿真系统的发展也促使了模拟集成电路建模方法的改变。仿真器的技术特征从器件特征函数与性能指标计算算子整合在同一模型中，发展到模型与仿真计算算子分离的技术特征[16-18]。目前，常用的系统级建模方法是以 VerilogA 模型[19-21] 和 ModSpec 模型[22] 为代表的方法，其核心是直接给出电路的端口方程或者性能指标方程，然后直接计算其性能指标。但是，VerilogA 在某些部分使用解析的表达式，无物理实现对应的函数和计算项和初始值，在模型复杂度和准确度方面不及 SPICE。在某些部分依旧使用结构表达，模型编写需要遵守若干最佳实践规范，否则难以正确仿真。ModSpec 模型将所有计算归结为可组合的代数差分方程，通过分离方程变量建立计算过程，但是对数字系统的离散参数适用性差。

近年来，在数字电路设计领域方兴未艾的敏捷设计方法，为数字电路设计带来了快速设计迭代与复杂约束条件下性能指标的快速探索等长期以来设计人员迫切需要的功能，极大提到了设计效率和设计成果的产出速度。针对数字电路

敏捷设计的高层次 Chisel 模型[23]，着重于系统整体的功能、数据与连接性，通过 FIRRTL 中间语言[24-25]实现了同标准 Verilog 模型的互相转化。实现既可以在系统层面使用高级语言开发提高处理复杂系统的能力，也可以在RTL层面使用电路语言开发确保模块可综合可实现。在基于元器件的数字电路结构设计中，已经有许多有益的尝试。例如，以 RC 电路模型和仿真数据为基础的，针对逻辑门的延迟指标的 Logical Effort 模型[26-27]，以及考虑了互连线负载的 Extended Logical Effort 模型[28]。该模型支持多种版图尺寸参数和多工艺角的延迟指标计算，并且支持延迟优化。

但是在模拟电路设计领域，敏捷设计并未大规模被设计人员采用。主要原因是，现有仿真系统的使用方法落后于业界需求和最佳实践，从而导致这些工具并不俱备对于快速原型设计的支持能力。比如，SPICE 仿真系统中，用户层面只能使用网表表达电路结构，同时在仿真计算时作为对于电路器件的调用指示。而有关性能指标和器件特征的计算算子组成的计算模型，封装在预先实现的模型文件中，无法被后续的设计人员按实际需求更新或者调用。即便是在 SPICE 仿真系统上发展起来的 Gnucap[16] 和 Qucs[17], 也采取了类似的模型编制和使用方案。Xyce 仿真环境[18]则在将器件模型与仿真计算算子分离方面进行了有益尝试，但是仍然需要设计人员直接编写计算算法，需要较高的编程能力，在实际生产环境中不具有通用性和便捷性。

不仅如此，现有的系统级模型中缺乏对于复杂约束条件的表达方法，相应地也缺乏以合理方式在考虑复杂约束条件下调度仿真器进行仿真的方法，对外在运行条件参数的处理相对简单。面对复杂约束条件的仿真要求，虽然电路规模并没有明显扩大，但是需要获得的数据点的数量增长速度远高于代表规模的拓扑中元件数量的增长速度。在 VerilogA 的语法规范中，涉及仿真规划与调度的关键词与 Verilog 语言使用的机制相同，以 if-else 为核心。但是仅仅依靠在现有模型文件中采用简单的 if-else 结构难以准确高效表示多重复杂的约束条件。并且，此种表示方法最大的问题是外在参数直接体现在电路模型的端口方程中。将外在约束条件与电路内在性能指标的计算方程混合在一起，而又不属于电路自身的参数，客观上造成了二者的深度耦合，解耦困难，导致难以在 module 作用域外部执行条件判断。还会造成了此类系统级仿真计算中，代数微分方程求解系统的额外负担和迭代操作，以及在算法那内存管理方面的混乱局面。

3 模拟集成电路系统级建模

现有研究主要缺乏对于电路系统级模型的系统化建模方法，以及对于在建模过程中有效整合复杂外在约束条件的方法。从方法论的角度来说，虽然所使用的要素以及模型建立的层次不同，后续可用的自动化生成方法也不尽相同，但是针对同一个事物建立的模型可以相互转化。尤其是当某个模型依赖的技术路线无法完全解决问题时，可以利用模型转化在另一个技术路线下尝试解决。这些问题都对更加高效敏捷的模拟电路自动化建模提出需求，直接促进了模拟集成电路的模型生成方法的相关研究。

为支持模拟集成电路设计中的电路结构和参数探索，尤其是清晰表达复杂约束条件并为电路设计提供评估机制，本文提出一种模拟集成电路系统级模型的建模方法。系统是由一系列互相连接或互相依赖的组件组成的一个整体。系统级模型是一系列相互联系并相互制约的反映目标电路系统某一方面特征的组件组成的模型。模拟电路的系统级建模，就是将模拟电路每个电路元件和结构以及设计约束都视作完整系统中的组件，赋予其特征并使用参数表示，然后通过系统工程方法得到电路的系统级模型。

当模拟集成电路的模型用作目标电路系统的设计规格说明时，模型会同时体现设计意图和设计约束，但对于电路设计实现的细节则不做过多说明。

本文提出的模拟集成电路系统级模型，组件以各种参数为基本组成单元，以参数关系及其约束关系为核心构造系统级模型，使用数学公式表示参数之间的关系。建模过程从现有电路结构提取有关参数和关系，在适当的建模过程中将电路拓扑结构作为与其同等地位的节点引入建模所得模型，抽象电路拓扑结构限制。得到的系统级模型并不依赖于作为来源的电路结构，而是将拓扑也作为一种参数的约束条件，只保留层次化的输入输出端口和特定参数关系。可以支持对于电路端口特性的多样化需求，以及对于电路在复杂的外在约束条件的下，更系统化的关键性能指标获取机制和更广阔设计空间的探索方法。

需要注意的是，本文提出的模拟集成电路系统级模型突出了参数及其之间的关系。这些参数来自于设计的标的和设计人员提出的规格和约束，但是不可避免地不能涵盖目标电路系统的所有方面。对于需要仿真获取的目标参数保留所需的仿真信息，可以在模型中建立更简单但是不失精度的函数方程。系统级建模方法与现有基于器件模型或行为级模型的建模方法并不冲突。在系统级模型中，现有行为级模型，RTL模型和电路级模型等层次的模型为系统级模型提供参数和参数之间关系的确实证据。在系统级模型中，所需的数据通过提供接口参数，调用现有仿真机制，简化了电路性能指标计算算子库的构建工作，提升了算子库的构建效率和质量。

面对复杂多样的设计约束条件，模拟集成电路系统级模型将约束条件引入模型，将异质信息同质化统一处理，从而解耦电路性能指标矩阵与环境参数矩阵。系统级模型提供对于目标电路系统整体拓扑和多类型约束条件的合理抽象，并且保证其数据精度在合理范围之内，为后续模型计算提供统一的信息视图。

系统级模型自身可以运算，可以脱离给定的仿真和计算环境的限制。但它不同于以仿真手段预测电路行为。它支持探索在仿真计算环境中不便直接表示的计算路径规划问题，模型信息完整性问题，设计可满足性问题，和潜在约束是否矛盾的判断等工作。

在本文提出的模拟集成电路系统级建模的数据获取和计算的技术层面，因所涉及参数之间关系的逻辑链条分支多且通常较长，并且约束条件中引入判断过程，因此，系统级建模将仿真环境作为数据获取的必要手段。系统级模型的建立过程需要电路仿真和电路分析，但是不以仿真作为闭环的终点，而是通过仿真获得数据实现模型自身的迭代。系统级建模工具充分利用现有仿真技术体系获得必要数据，仿真环境完全依赖于数据需求，构成包含在建模模型中的一个被调

用模块。这些基础数据作为计算和逻辑判断中的逻辑不变量，在保持接口参数一致性的前提下，可以为整体的系统级建模提供高精度的数据。

需要注意的是，采用模型映射技术可以将本文提出的系统及模型映射到领域专用仿真环境或多领域仿真环境中。例如，模拟集成电路系统级建模的语言要素可以转化成为普通系统建模语言的一种特定领域构造型，从而支持系统级模型的仿真。但是需要说明的是，领域构造型并没有增加语言要素自身的语义特性。

系统级建模包含三个方面的工作：建模语言，建模方法和建模工具。以下逐一说明。

3.1 模拟集成电路系统级模型建模语言

为构建模拟集成电路系统级模型，本文首先给出用于建模的DSL语法规范。本文提出的模拟集成电路系统级建模语言的基本形式为Name-Value-Pair (Name Value Pair, 名称-值对)。在Name-Value-Pair中，使用 $<=$ 连接名称和值。

该语言对建模中需要给定的 Node、Variable、Function 和 Constraint 等语言要素给出形式化定义和相应的语法规则。模型至少要包含一个 Node。

Node 是系统级模型的基本组成单元。从语义上来说，Node 是一个包含了若干 Variable、Function 和 Constraint 定义的实体。Node 可以被嵌套。其中的元素可以通过属主引用。从系统整体来看，Node 可以不受到电路拓扑结构作为组件带来的组件边界限定。对于模拟集成电路来说，电路的天然层次化结构可用通过 Node 的嵌套和引用实现层次化电路描述。

Variable 表示系统级模型中的变量或常量，它可以是 Node 的属性，也可以是全局属性。从语义上来说，Variable 记录了属性具体的信息，包括但不限于：执行所处计算环境的命令行、返回并更新指定 Variable。Variable 在使用时支持版本化管理，应对不同设计条件和约束的组合。

Function 表示属性之间的关系。在计算中，它支持符号计算，支持按需要调整和查询输入输出组合。从语义上来说，Function 是多入单出结构的。它实质上只定义属性关系的输入和输出，并不需要定义实际的计算过程。

Constraint 特别表示属性之间的约束关系，是 Function 的一种。从语义上来说，Constraint 对 Node、Variable 或 Function 设置限定条件。Constraint 本质上是无限制的，它可以嵌入到任何类型的单元上，Constraint 的参数可以没有确切物理含义。对于模拟集成电路来说，Constraint 也可以记录电路拓扑结构信息，此时，拓扑结构作为一种限定条件。

此处总结了模拟集成电路系统级建模语言的BNF (Backus-Naur Form, 巴科斯范式)语法规则，如图3所示。

使用模拟集成电路系统级建模语言编制的模型文本由定制语言编译器编译。编译器可以完成语言解析，命令调用和数据处理和更新等工作。但该编译器不会将模型直接编译为可用于计算的二进制代码。

```
grammar anasysmod;

prog: stat+ ;

stat:           nodedefsec     # nodedefsecglobal
        |       funcdefsec     # funcdefsecglobal
        |       consdefsec     # consdefsecglobal
        |       vardefsec      # vardefsecglobal
        |       refsec         # refsecglobal
        ;

nodedefsec:        'node' nodename '{' nodebody* '}'  ;

nodebody:       nodedefsec      # nodedefin
        |       funcdefsec      # funcdefin
        |       consdefsec      # consdefin
        |       vardefsec       # vardefin
        ;

funcdefsec:    'func' funcname  '<=' '(' funcparm (',' funcparm)* ')' ;

consdefsec:    'cons' consname  ('<=' value )? ;

vardefsec:     'var' varname  ('<=' value )? ;

refsec:         nodename '.' refname  ('<=' value)? ;

funcparm:       refsec          # funcparmref
        |       varname         # funcparmvar
        ;

refname:        SYMB+ ;
funcname:       SYMB+ ;
nodename:       SYMB+ ;
varname:        SYMB+ ;
consname:       SYMB+ ;

value
    : NUMBER (SYMB)?                   # valuenum
    | STRING                    # valuestr
    ;

COMMENT
    : '//' ~ [\r\n]* NEWLINE -> skip
    ;

NUMBER: '-'? INT ('.' [0-9] +)? EXP?;
fragment INT: '0' | [1-9] [0-9]*;
fragment EXP: [Ee] [+\-]? INT;

NEWLINE :   '\r'? '\n' -> skip;
WS      :   [ \t]+ -> skip ;

SYMB: [a-zA-Z][a-zA-Z0-9_]* ;

STRING : '"' (ESC | .)*? '"' ;
fragment ESC: '\\' [btnr"\\] ;
```

图 3: 模拟集成电路系统级建模语言的BNF语法规则

3.2 模拟集成电路系统级模型建模方法

模拟电路系统级建模整合多方面的领域设计知识和设计经验。根据本文给出的模拟集成电路系统级建模语言，模拟集成电路系统级建模方法围绕参数之间的解耦展开。

系统级模型的建模信息来源，一方面是有关系统规格的说明和外在运行环境的约束条件；一方面通过描述物理系统的特征获得有关信息，包括待建模电路的图结构描述，由器件特性和仿真需求等确立的性能指标的计算算子和方程等。提出设计约束需要有基础数据作为支持，但是设计约束是否满足或者约束是否合理，需要后续仿真计算进行验证。

在本文提出的模拟集成电路系统级模型建模方法中，蕴含在计算关系中的参数耦合，比如设计规格约束信息，可以通过限定条件转变为逻辑条件判断处理，从而可以将参数与限定分离表示。这些计算关系体现的是系统层面的内容，其来源包括：基于曲线拟合近似的半解析模型，基于仿真数据拟合的数据驱动模型，基于电路分析的解析模型，基于已有端口模型语义转换等。规格与需求的变动，也都体现在限定条件的改变上。

参数之间的耦合既蕴含在计算关系中，也出现在跨越物理元件边界的情形中。电路的拓扑结构是电路设计的一类重要内容。从模拟集成电路的拓扑结构来看，各个组成物理元件是时刻连接的。在建模时，电路元件间的连接点和元件均定义为 Node。依据元件和连接点之间的连接情况定义相应的 Function 组件。元件连接之间的限定条件是KCL (Kirchhoff's Current Law, 基尔霍夫电流定律)和KVL (Kirchhoff's Voltage Law, 基尔霍夫电压定律)。由这些限定条件可以构造相应的 Constraint 组件。在系统级模型中，拓扑的物理组成和电路结构划分不作为强制的限定条件。不同于完全基于拓扑结构边界的参数和约束分配式建模，参数或由参数组成的约束关系不必须被分配给特定的电路元件。而由于跨越物理元件边界导致的参数耦合通过将物理元件边界定义为限定条件，使得电路拓扑结构与参数计算关系解耦。整体上看，本文提出的模拟集成电路系统级模型建模方法，重构电路性能指标计算的核心流程，对性能指标的计算算子进行系统化重组和优化。

本文提出的模拟集成电路系统级建模语言构造的系统级模型以图Graph的基本理论和方法为基础。在集成电路设计中，Graph已经被证明具有重要的基础性作用，是许多电路设计和优化方法采用的基础数据结构。在数字电路设计中广泛用于逻辑综合，布局布线，以及版图提取等方面的研究。在使用Graph作为描述电路的数据结构方面，有关集成电路提取中子图同构的研究，为基于Graph的电路形态改变的计算提供了有益的思路和参考。

本文采用Graph的数据结构存储，用来记录电路拓扑的结构，也用来记录复杂限定条件的相互关系。使用Graph构建以模拟电路设计和仿真中性能指标及其相关参数关系为核心的计算流图。其中包含了参数关系的计算流程，参数之间的限定条件，也包含了模拟电路本身的拓扑和结构信息，同时调用现有计算和仿真机制，为计算流图中各个参数提供必要的数据。

特别的，有关限定条件的子图是一种约束图（Constraint Graph）。约束图

是逻辑约束关系的一种图表示。在约束图中，节点表示了约束的来源和约束的对象，边表示约束的指向，边的权重表示约束的具体内容。将这样的约束图与电路拓扑结构的图进行比较可以发现，虽然在图中，不同类型的节点和边具有不同类型的计算规则和推导规则，但是两种图中的节点均表示某种对象，而边则均表示对象之间的关系或者联系。因此，可以将约束图与电路的图融合起来，也就是在系统级建模过程中，将限定条件引入建模过程。

从系统级建模的角度来看，复杂限定条件与模拟电路的宏模块及基本元件一起构成了图的两类节点与边。参数关系也可以表征拓扑信息和结构特点，从而将拓扑和结构问题封装在计算图的节点内部，将异质要素进行同质化处理，既实现了不完全依赖于电路拓扑结构的计算，又实现了高效的模拟电路系统级建模。在计算流图中，性能指标和相关参数的计算，与电路的拓扑和结构信息具有不同的图上元素的特征，在图上是分离表示的，从而在形式上实现了性能计算与电路结构之间的解耦。同时解决了领域知识的形式化表达映射到计算流图的方式和方法问题。最终获得的系统级模型的图数据结构是一种混合了两种节点和边的图，将模拟电路系统级建模问题转换为一个计算流图的设计和计算问题。在这方面，基于 PERT 的层级化方法是处理计算流程图的基本方法。因为该系统级模型是基于逻辑依赖关系建立的，因此凡是可以处理逻辑关系的系统也都可以处理该系统级模型。

模拟集成电路领域知识、系统级建模语言要素和混合计算图要素的对应关系如表??所示。

表 1: 模拟集成电路领域知识和系统级模型要素对应表

领域设计知识	系统级建模语言要素	混合计算图要素形式	在系统级模型中的特点
电路拓扑信息	Node 或 Constraint	节点或子图	恒成立的约束条件, 作为其他命题的前提条件
基于拓扑信息提取出的其他信息	Variable	节点	后续计算独立于拓扑信息
电路设计约束	Constraint	节点	约束也是参数之间的一种关系
设计参数之间的关系	Function	计算流图的边，或混合计算图的节点	被视为一个命题, 记录关联性, 可以代表复杂运算过程
仿真计算和基础数据	Variable	节点	可以被仿真调用和更新

3.3 模拟集成电路系统级建模软件工具

本文提出的模拟集成电路系统级模型的建模工具基于领域驱动设计（Domain-driven Design）的程序设计方法[29-30]。领域驱动设计方法采用了复杂性思想作为设计方法论基础，梳理出业务域模型并建立领域专用语言，通过持续集成的方式快速迭代，可以充分利用领域模型良好的需求适应能力和架构演化能力，以敏捷开发方式开发适用于特定场景的软件工具。

经过前期调研以及近期查询，模拟集成电路系统级建模工具的实现方式，一种是作为一个功能模块在工具软件中体现，此类软件多数以电路拓扑设计软件或仿真计算环境为主。另一种体现了系统级设计理念的设计语言为表现形式，比如 VerilogA 语言或 MAPP 设计框架。在上述两大类实现形态中，有关系统级直至电路级约束的表示和依赖关系计算，以及精度控制都依赖于各自工具的基础计算环境和程序语言给定的算子和算法。

目前，以统一语法形式进行带有逻辑判断计算的工具多数为规则引擎类工具，其主要问题在于鲜有以模拟集成电路和电路设计为背景进行开发的工具，因此并没有考虑电路设计和电路计算的需求和特点，缺乏在同一模型层面以系统级建模内容为核心，统一表示并计算推断性能指标和约束条件的工具。

本工具着重于模拟电路系统级性能指标参数和约束条件的统一语法形式记录和计算，以电路设计中的参数关系和约束关系为中心，在建模语言和编译器设计中设计了多种直接引入拓扑数据和仿真数据的方法，较通用规则引擎具有更大的适用性和表达上的优势。在电路拓扑和仿真计算方面直接应用和接入现有成熟的系统和工具，并根据建模使用的参数，根据接入的系统和工具的要求，进行了计算过程的优化和简化。在数据精度上应可以保持一致，在计算效率上不劣于接入系统的原有效率。本工具的系统结构如图4所示。

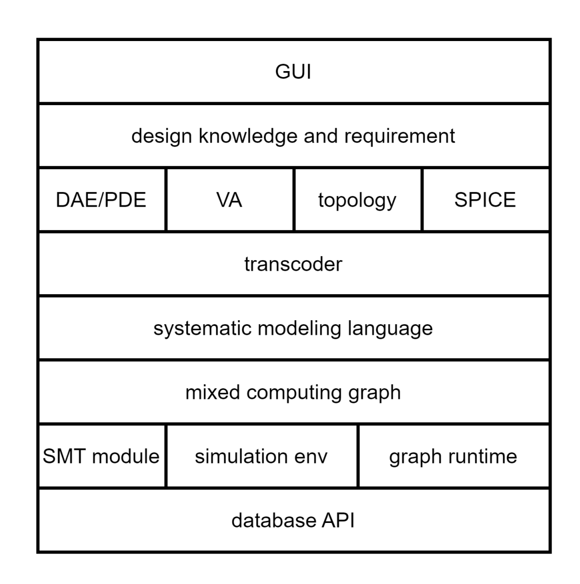


图 4: 模拟集成电路系统级建模工具层次结构

该工具的输入是待建模电路的某种表示，它使用的表示语言既可以是 SPICE

类型的仅由电路元件描述组成的网表文件，也可以是使用 VerilogA 描述的电路的端口特性以及性能指标计算特征。该工具的输出是对待建模电路及其运行环境和约束条件的一种统一的系统化的描述，并且可以被进一步被其他工具转化成现有仿真系统可以识别的描述语言，供仿真使用。用户可以通过工具给出的信息去探索系统的模块、约束条件的关系、参数计算函数等，进而对待设计的模拟集成电路作为一个系统的行为和结构进行逻辑分析和验证。

该建模工具采用 C++ 语言编制，并使用 Qt 作为图形前端。它的软件界面如图5所示。

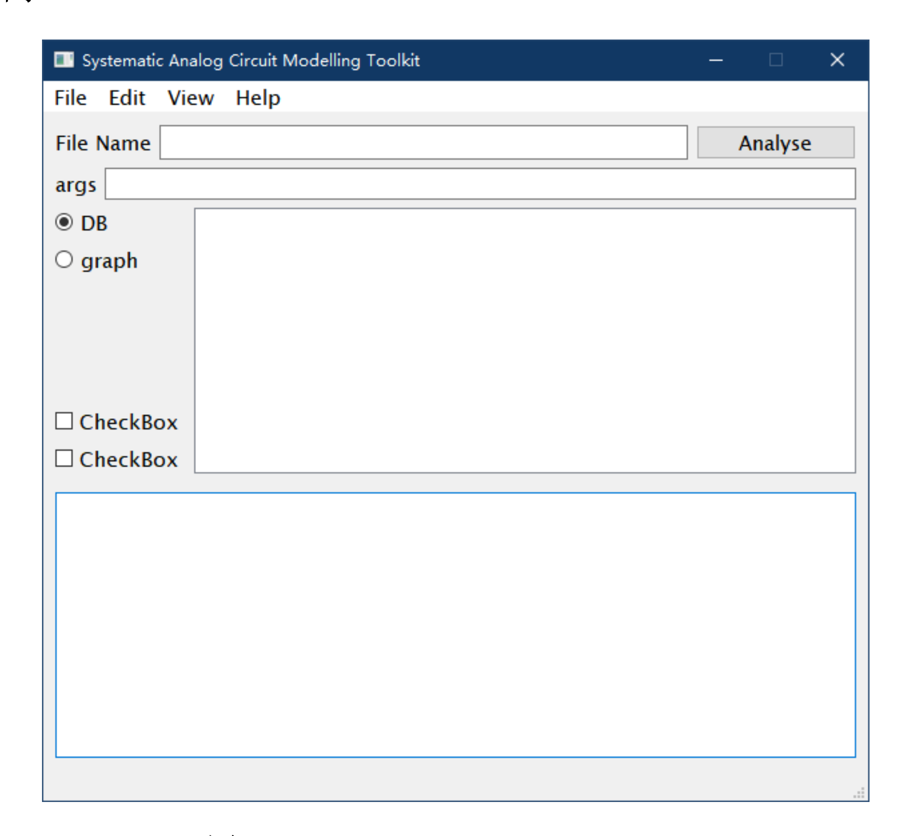


图 5: 模拟集成电路系统级建模工具界面

3.4 模拟集成电路系统级建模案例

此处给出的模拟集成电路系统级建模案例的问题来自于《CMOS 模拟集成电路版图设计：基础、方法与验证》[31]。该问题的需求是基于给定的电路拓扑结构和元件参数，根据设计规格确定其余元件的取值参数。该电路拓扑的结构如图6所示。

此处重新抄录给出的设计指标的参数并简要总结其设计过程。

- 增益：$A_v = 10$
- 带宽：$200MHz$
- 负载：$1pF$

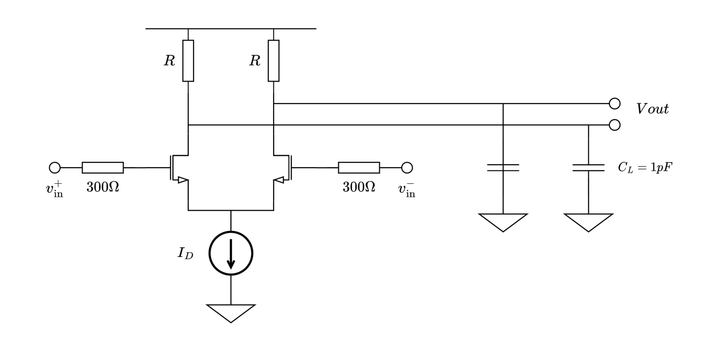


图 6: 差分放大器电路拓扑结构图

- 源输入阻抗：300Ω

通过 SPICE 仿真，可以建立基于 g_m/I_d 的参数关系。通过对不同晶体管长度 L 进行扫描，可以得到 $f_T \sim g_m/I_d$，$g_m r_o \sim g_m/I_d$，$I_D/W \sim g_m/I_d$ 等关系。因为采用了基于 g_m/I_d 的参数关系数值曲线图，数值曲线是基于不同的晶体管长度来绘制的，因此需要选定一个适当的 L。这里的选择依赖于一系列设计规格的约束。

在差分放大器中，主极点与负载电阻与负载电容有关，粗略的关系为：

$$\omega = \frac{1}{RC} \tag{1}$$

因此首先可以确定负载电阻 R 的数值。

根据增益与负载电阻之间的关系：

$$A_v = g_m R \tag{2}$$

可以得到跨导 g_m 的平均值。

输入端的 MOS 管自身引入的 C_{gs} 也会导致系统存在一个极点，由输入源阻抗可以得到 C_{gs} 的近似值。

此时，主要的极点都已经经过了分析和计算，可以就整体的放大器带宽的截止频率 f_T。它可以由下面的关系得到：

$$f_T = \frac{1}{2\pi}\frac{g_m}{C_{gs}} \tag{3}$$

根据 $g_m r_o \sim g_m/I_d$ 的数值曲线图，它首先要保证本征增益处于一个相对合理的范围。最终的设计结果证明选取的晶体管长度达到的这个设计约束的要求，并且保证 g_m/I_d 是一个适当的值。

此后就可以根据 f_T，查询确定 g_m/I_d 的数值。它是后续查询曲线的横坐标数值。继而可以立刻得到对应于特定 g_m 和 g_m/I_d 的晶体管的 I_d 的数值。此处需要根据差分拓扑的特征将 I_d 扩大为计算值的 2 倍。然后，再次查询 I_D/W 与 g_m/I_d 的关系曲线，就可以最终确定所需的晶体管的 W 值。

上述过程可以使用系统级模型语言建模并使用建模工具绘制出混合计算流图，如图7和图8所示。

diffamp.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
node diffamp {
var L <= 0.22
var av <= 10
var fband <= 200MHz
var CL <= 100pF
var rs <= 300
var R
func R <= (CL, fband)
var gm
func gm <= (av,R)
var Cgs
var freq2 <= 2GHz
func Cgs <= (rs, freq2)
var ft
func ft <= (gm, Cgs)
var gmID
func ID <= (gm, gmID)
func gmID <= (ft, L)
cons fttbl <= "sh -c echo ft.gmid.L.CSV"
cons gmrotbl <= "sh -c echo gmro.gmid.L.CSV"
cons IDWtbl <= "sh -c echo idw.gmid.L.CSV"
cons topo <= "sh -c spice -in diffamp.cir -o topo.cons"
cons Lrange <= "L>0.18 AND gmID > 10"
}
```

第 6 行，第 14 列 100% Windows (CRLF) UTF-8

图 7: 差分放大器系统级模型

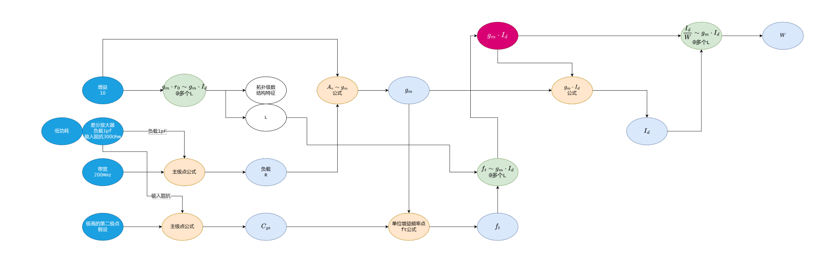


图 8: 差分放大器系统级模型的计算流图

根据混合计算流图，可以很容易得知该设计过程是否复合设计约束且是否可行。事实上，计算流图附加上限定条件可以成为判断逻辑可满足性是否成立的判断图。具体的，对于通过系统级模型，如果将系统级模型中的 Function 节点均作为逻辑判断的“与（AND）”节点，那么立刻可以得到有关设计过程的可行性判断图，如图9所示。

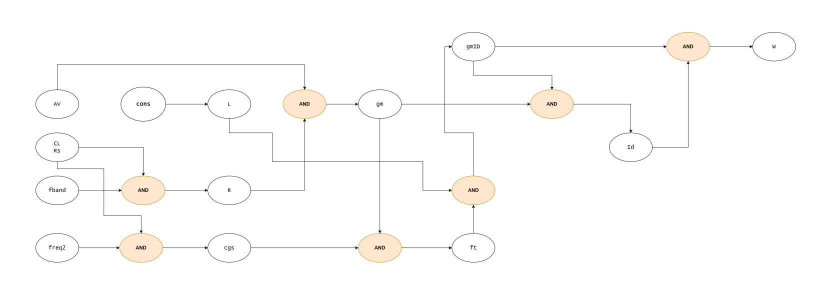


图 9: 差分放大器设计可行性判断图

明显可见，该设计满足设计约束条件和设计规格，并符合参数计算条件，是可行的。

4 结果和讨论

建模和仿真是密切联系且互相成为依赖的两个研究领域。系统级模型的发展回应了在仿真和建模研究中的迫切需求和不足。相应的，仿真工具的发展也会促进系统级模型与仿真环境进一步整合，共同促进模拟集成电路的设计和验证工作。本文分析了模拟集成电路建模的现状以及系统级模型的特征和建模需求。随后本文给出了模拟集成电路建模的语言、建模方法和建模工具，并给出建模案例。

未来，在本文提出的模拟集成电路系统级建模方法基础上，一方面需要进一步研究如何丰富建模语言，规范语言要素的意义和使用方法，一方面也要研究如何给出建模模板，令系统级模型覆盖更多典型的电路和使用场景，提高其适用性。此外，对于仿真系统的需求和整合也是未来工作的研究重点，包括研究内嵌仿真环境和计算环境为模块的方法，并提高参数的版本化管理能力，同时尝试在图计算框架下引入图分割、图内嵌和并行计算等技术，提高混合计算图生成和逻辑条件计算效率。

5 作者贡献

路卢　提出研究思路和框架，收集整理文献和资料，提出模拟集成电路系统级模型语言和建模方法，完成软件工具的编制和实验，完成全篇论文的文字写作和修订。

参考文献

[1] DELLIGATTI L. SysML distilled: A brief guide to the systems modeling language[M]. 1st ed. Addison-Wesley Professional, 2013.

[2] JANCKE R, SCHWARZ P. Supporting analog synthesis by abstracting circuit behavior using a modeling methodology[C/OL]//2006 IEEE International Symposium on Circuits and Systems. 2006: 4 pp.-1474. DOI: 10.1109/ISCAS.2006.1692874.

[3] H. AGARWAL Y S C A N S K, J. P. Duarte, HU C. BSIM6.1.0 MOSFET Compact Model[M]. http://www-device.eecs.berkeley.edu/bsim/?page=BSIM6 LR, 2014.

[4] PSP release: Level 103 v3.0, 2014.[M]. http://psp.ewi.tudelft.nl, 2014.

[5] RAKHEJA S, ANTONIADIS D. MVS Nanotransistor Model (Silicon)[R]. 2014.

[6] CORAM G. MOS Model 11 Level 1101[M]. http://www.designers-guide.com/VerilogAMS.

[7] NAGEL L W. SPICE2: A computer program to simulate semiconductor circuits[D]. EECS department, University of California,Berkeley, Electronics Research Laboratory Memorandum no.ERL-M520, 1975.

[8] PEDERSON D O, Sangiovanni-Vincentelli A. SPICE 3 Version 3F5 User's Manual[M]. Dept. EECS, Univ. California, Berkeley, CA, 1994.

[9] NENZI P, VOGT H. Ngspice Users Manual Version 26 (Describes ngspice-26 release version)[M]. 2014.

[10] GIELEN G G E, RUTENBAR R A. Computer-aided design of analog and mixed-signal integrated circuits[J/OL]. Proceedings of the IEEE, 2000, 88(12): 1825-1854. DOI: 10.1109/5.899053.

[11] HAN D, CHATTERJEE A. Simulation-in-the-Loop Analog Circuit Sizing Method Using Adaptive Model-Based Simulated Annealing[C]//4th IEEE International Workshop on System-on-Chip for Real-Time Applications,. 2004: 127-130.

[12] SILVEIRA F, FLANDRE D, P. G. A. Jespers. A gm/ID based methodology for the design of CMOS analog circuits and its application to the synthesis of a silicon-on-insulator micropower OTA[J/OL]. IEEE journal of solid-state circuits, 1996, 31(9): 1314-1319. DOI: 10.1109/4.535416.

[13] JESPERS P G A, MURMANN B. Systematic design of analog CMOS circuits: Using pre-computed lookup tables[M/OL]. Cambridge: Cambridge University Press, 2017. DOI: 10.1017/9781108125840.

[14] LIN P M. Symbolic network analysis.[M]. New York Elsevier, 1991.

[15] SHI C J, TAN X D. Canonical symbolic analysis of large analog circuits with determinant decision diagrams[J/OL]. IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems, 2000, 19(1): 1-18. DOI: 10.1109/43.822616.

[16] A.Davis. The gnu circuit analysis package[M]. http://www.gnu.org/software/gnucap, 2006.

[17] BRINSON M E, JAHN S. Qucs: A GPL software package for circuit simulation, compact device modelling and circuit macromodelling from DC to RF and beyond[J]. Int J Numer Model Electron Networks Devices Fields, 2009, 22: 297-319.

[18] E. R. KEITER R L S H K T J C D A F T S C R P P C E W E A T V R, T. Mei. Xyce parallel electronic simulator users' guide, Version 6.0.1.[R]. Technical report, Raytheon, Albuquerque, NM; Sandia National Laboratories (SNL-NM), Albuquerque, NM (United States), 2014.

[19] CORAM G J. How to (and how not to) write a compact model in Verilog-A[C]//Proceedings of the 2004 IEEE International Behavioral Modeling and Simulation Conference, 2004. BMAS 2004. IEEE, 2004: 97-106.

[20] MCANDREW C C, CORAM G J, GULLAPALLI K K, et al. Best practices for compact modeling in Verilog-A[J]. IEEE Journal of the Electron Devices Society, 2015, 3(5): 383-396.

[21] VERILOG HDL. Verilog-AMS Language Reference Manual[J]. Eda-Stds. Org, 2014.

[22] AMSALLEM D, ROYCHOWDHURY. J. ModSpec: An open, flexible specification framework for multi-domain device modelling.[C/OL]//Computer-Aided Design (ICCAD), 2011 IEEE/ACM International Conference On. IEEE, 2011: 367-374. DOI: 10.1109/ICCAD.2011.6105356.

[23] BACHRACH J, VO H, RICHARDS B, et al. Chisel: Constructing hardware in a scala embedded language[C/OL]//DAC '12: Proceedings of the 49th Annual Design Automation Conference. New York, NY, USA: Association for Computing Machinery, 2012: 1216-1225. https://doi.org/10.1145/2228360.2228584.

[24] IZRAELEVITZ A, KOENIG J, LI P, et al. Reusability is FIRRTL ground: Hardware construction languages, compiler frameworks, and transformations[C/OL]//2017 IEEE/ACM International Conference on Computer-Aided Design (ICCAD). 2017: 209-216. DOI: 10.1109/ICCAD.2017.8203780.

[25] LI P S, IZRAELEVITZ A M, BACHRACH J. Specification for the FIRRTL language: UCB/EECS-2016-9[R/OL]. EECS Department, University of California, Berkeley, 2016. http://www2.eecs.berkeley.edu/Pubs/TechRpts/2016/EECS-2016-9.html.

[26] SUTHERLAND I, SPROULL R F, SPROULL B, et al. Logical effort: Designing fast CMOS circuits[M]. Morgan Kaufmann, 1999.

[27] MORGENSHTEIN A, FRIEDMAN E G, GINOSAR R, et al. Unified logical effort—A method for delay evaluation and minimization in logic paths with $ RC $ interconnect[J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2009, 18(5): 689-696.

[28] 路卢, 彭思龙. 使用扩展逻辑效力的逻辑路径尺寸优化方法[J]. 计算机辅助设计与图形学学报, 2009, 21(11).

[29] EVANS E, EVANS E J. Domain-driven design: Tackling complexity in the heart of software[M]. Addison-Wesley Professional, 2004.

[30] VERNON V. Implementing domain-driven design[M]. Addison-Wesley, 2013.

[31] 陈铖颖, 范军, 尹飞飞, 等. CMOS 模拟集成电路版图设计：基础、方法与验证[M]. 机械工业出版社, 2022.